Statement of John Hale
Assistant Professor of Computer Science and
Director, Center for Information Security,
The University of Tulsa
Before the Committee on Government Reform
U.S. House of Representatives

Oversight Hearing on Security and Privacy in Peer-to-Peer Networks

May 15, 2003

Mr. Chairman, Ranking Minority Member Waxman, and Members of the Committee, thank you for giving me the opportunity to testify before you today on a topic that is of growing concern to the computer security community, to American businesses and schools, and, in fact, to anyone that uses the Internet.

I am an Assistant Professor of Computer Science at the University of Tulsa, and serve there as the Director of its Center for Information Security. As an information security researcher and an educator, I have watched P2P technology make a startling transition from the backwaters of computer science to pop culture in mainstream society.

This March, Sharman Networks hit the 200 million mark for downloads of its popular Kazaa Media Desktop. Over the past two years, the active host count at any given time in the Gnutella network has ranged from 100,000 to 500,000. P2P software is installed on computers in homes, businesses and schools across the world. P2P networking has grown faster than the Internet itself, and has reached a much broader audience at this stage of its development.

Part of the attraction of P2P networks is their dynamic nature. P2P technology creates flexible *ad hoc* networks that span the globe, connecting end users in a peer-wise architecture that is both resilient and efficient. Search engines built into P2P clients are powerful and intuitive. They put a staggering volume and variety of digital content at a user's fingertips.

But there is a downside to placing such a potent technology in the hands of novice users. A P2P client can turn a computer into a server, exposing it to a new range of threats. Installation and operation is so easy that most do not fully appreciate the risks.
And deceptive practices of the purveyors of P2P file sharing software who are trying to stay one step ahead of copyright owners and network administrators have made the situation much worse.

Spyware and Adware

The prevalence of embedded spyware and adware in P2P clients is but one example. Spyware monitors user behavior and tracks web browsing habits. The information collected by spyware is typically sold to companies and/or used by adware to conduct targeted web marketing. Based on an individual's browsing patterns, adware opens web pages promoting a particular product or service.

P2P developers bundle spyware and adware in their clients to generate revenue. One P2P company maintains that its embedded spyware is "integral" to the operation of their product. Of course, there is no inherent functional dependency between advertising and file sharing. In fact, lightweight implementations of P2P software have been developed that leave the spyware out. "Integral" means that the P2P software has been deliberately engineered so it will not function without the spyware active.

Spyware and adware are, by construction, difficult to detect and may be impossible to disable or remove from a client. Common tactics include hiding in system folders and running in the background from system startup. Amazingly, some spyware components remain on a system long after the original application is removed, and will even embed themselves in a host despite an aborted installation of the carrier application.

Spyware not only poses a threat to user privacy, it can also create additional vulnerabilities on a user's system. Spyware products embedded in the most popular P2P clients download executable code without user knowledge. Even if the code is not malicious, it may contain flaws that render a system open to attack. The clandestine nature of the software makes detection and remediation extremely difficult.

Circumventing Security

P2P software is commonly designed to circumvent network security services. Enterprises and institutions wishing to stem the tide of media piracy on their networks often find that P2P file sharing traffic is disguised as or hidden amongst normal network activity. Techniques such as tunneling, port hopping and push requests make it difficult to detect and filter P2P traffic. That is their intent; to foment user participation in spite of an enterprise's security policy. One consequence (intended or not) is that these techniques dramatically weaken an organization's security posture.

Tunneling embeds P2P messages within another protocol so that they blend in with other traffic, making them more difficult for firewalls and filters to detect. A common scheme is HTTP tunneling, in which P2P communications are disguised as web browsing traffic. This variation is popular because web traffic is so common and typically travels freely across enterprise networks. To this end, tunneling not only helps violate a network security policy by enabling forbidden applications but also expands the network security perimeter in ways unknown and unpredictable to system administrators.

Another commonly used trick is for P2P clients to vary their communication ports – a technique called port hopping. This thwarts blocking and scanning software that identifies network services based on well-known port assignments. Port hopping is built into the latest versions of the most popular P2P clients – and there is no reason for it other than to allow network software clients to avoid detection.

Developers of the Gnutella protocol devised a special solution that permits clients to circumvent firewalls configured to block its file request messages. In this scheme, a 'push-request' message is sent through the Gnutella network to the system behind the firewall, which then knows to initiate a file upload to the requesting host. So instead of a client 'pulling a file to it,' it asks the serving system to 'push the file out.' To the user, the net effect is the same – they get the file – but to the firewall, which usually has looser restrictions on out-bound traffic, it makes all the difference in the world. And once again, an enterprise's network security policy is violated.

Software Vulnerabilities

Another major concern is how software flaws in P2P networking clients can greatly increase the exposure in a network, leaving it vulnerable to intruders and hackers. All software has flaws, and some flaws create exposures that can be exploited to violate the security of a system.

Exploitable weaknesses in P2P software have been identified. Buffer overflow and cross-site scripting vulnerabilities were reported in early iMesh and Gnutella clients, respectively. P2P clients that use the Fasttrack protocol are known to be susceptible to Denial of Service attacks due to its client-to-client messaging architecture. Sometimes the shared files themselves enable an attack. MP3s contain special meta-data that in the past has been used to exploit buffer overflow vulnerabilities in media players. In this particular attack, a P2P network is simply a distribution mechanism for the malicious payload, but it is an incredibly effective one.

There is nothing special about P2P software that makes it inherently more flawed than other software. It is built for the same platforms and developed in the same programming languages as other computer and network applications. However, several factors conspire to make the risks induced by security vulnerabilities in P2P file sharing clients much more serious.

The first factor is that P2P clients engender massive *ad hoc* connectivity across organizational and enterprise domains. P2P file sharing networks are well beyond the administrative control of any one company or organization. A system running a P2P client may be behind a firewall, but it is exposed through the client to every user on that P2P network, regardless of their location. Simply put, P2P clients can dramatically amplify exposures to external threats.

A related factor deals with trust. P2P file trading networks are open environments that allow anyone to share files pseudo-anonymously. Trust in this circumstance is hard to come by. Users are connected to and download files from hosts they know very little about. In many cases, the P2P client itself is installed in a bootstrap process that downloads it from a peer on the network. P2P file sharing networks expose systems to untrusted hosts and software, and offer little in the way of protection.

Enterprise security management in the presence of contraband P2P file sharing software is a supreme challenge. The dynamic nature of P2P networks, the stealth tactics employed by the software and the tendency of individuals to hide its use makes a complete inventory of P2P clients on a large network virtually impossible. This again magnifies any security vulnerabilities because inventories are essential for security remediation processes. It is very difficult to address problems on a network if you cannot find the software that is causing them.

Worms and Viruses

No discussion of security threats to P2P networks is complete without covering the potential for viruses and worms. Viruses and worms are self-replicating code that may or may not contain a malicious payload. The difference between the two is that a virus typically requires some form of human participation to propagate while a worm can spread across a network without human intervention. Both are viable modes of attack in P2P networks.

A P2P virus needs a carrier file to contain its payload. The obvious choices are audio, video and executable files traded over the network. Buffer overflow vulnerabilities have already been exploited in media players by maliciously crafted MP3 files. A virus can leverage such a weakness to execute code that replicates itself in the shared folder directory of a user. The act of downloading an infected file spreads the virus to a new host.

The recent integration of executable content in media formats creates a richer entertainment experience, but also offers a limitless palette for viral code. I am reminded that e-mail attachments became the preferred mode of virus transmission after the introduction of active content in word processing documents and web pages. Scripting means you no longer have to break an application with a buffer overflow attack. Instead, you can exploit weak security policies and input validation processes to achieve the same effect.

Several so-called P2P worms have been documented. The Duload P2P worm may be the most sophisticated of these. This piece of malicious code copies itself into the system folder and alters the registry so that it always runs at startup. It then copies itself to several provocatively named files within a media folder which it exposes to the P2P network as a shared folder. Since Duload relies on a human to download, it really acts as a virus. A true P2P worm would have to exploit a flaw in a P2P client to propagate itself across a network.

The P2P viruses uncovered to date barely hint at the real potential of self-replicating code in P2P environments. Code Red, Nimda and Slammer – worms that targeted Internet web and database servers – provide much better insight. The Code Red Internet worm infected 359,000 Internet hosts within 14 hours, causing an estimated $2.6 billion in damage. The Nimda worm caused an estimated $590 million in damage and infected 2.2 million hosts. Comparatively, the Slammer worm only infected 200,000 hosts, but set new speed records, infecting 90% of the hosts vulnerable to it on the Internet in an astonishing 10 minutes.

Likewise, a self-propagating P2P worm could infect almost every host on the P2P network, crossing enterprise network boundaries with blazing speed. More importantly, the previously discussed obstacles to efficient remediation indicate that a P2P worm would have tremendous staying power, re-infecting unpatched hosts and infecting new ones as they came online.

There is a role for technology to play in addressing these problems. Tools and systems can be developed to better monitor and secure hosts running P2P clients. Of course technology is only one piece of the solution. Users must be made aware of the risks of participating in open P2P file sharing networks. Developers must be held accountable and live up to higher standards of integrity and transparency for the P2P software they build. Ultimately, P2P technology must shed its reputation as a tool for media piracy.

In a very real sense, peer-to-peer file trading software exposes individuals and enterprises to risks above and beyond those of other software. The technology itself is beautiful in its design, but developer and user practices conspire to create a dangerous operational environment. On its current evolutionary track, threats to security and privacy posed by P2P file sharing technology will get worse, not better. We cannot predict the next Code Red or Nimda. But if and when it strikes peer-to-peer networks, I hope we do not look back to this moment in time and see a missed opportunity to lead a promising technology out of a turbulent period in its development.